[Skills Test] Backend Engineer

Introduction

Thank you for taking the time to do this skill test. We know your time is valuable, and we really appreciate your commitment.

We expect you to spend around **3 - 5 hours** going through this test before submitting your progress. Be as thorough as you can, but every solution, even if incomplete, is valuable to us in understanding how you think and how you communicate.

When you are finished, please submit your solution in a **compressed archive (.tgz, .zip) via email**.

Good luck!

Goal

ℹ️ At Qonto, we have a wide variety of customers. Some of them are big organizations that need to perform many transfers. For example, paying the salaries of hundreds of employees at the end of the month.

Performing those transfers one by one would be painful and time-consuming for our customers, and for that reason we allow them to perform transfers *in bulk*.

Your mission is to **write a web service to receive bulk transfer requests from a single Qonto account** and:

1. **Verify the validity of the request**: whether the Qonto customer has enough funds for all the transfers in the request. If the customer does not have enough funds, the entire request should be denied;
2. If the request must be denied, return a 422 HTTP response;
3. Otherwise, add the transfers to the database, update the customer's balance, and return a 201 HTTP response.

💡Take care to consider the following:

- Ensure you **don't lose a cent** during processing,
- Assume the server will have **multiple load-balanced instances**,
- Consider that **any process can crash at any point** without warning, including the client; similarly, any network connection can glitch,
- Pretend you are using **PostgreSQL, MySQL, or a similar relational database**, but you can use the provided Sqlite database instead to get going faster,
- Ensure you **don't accept a request that the balance shouldn't allow**, and vice-versa.

Instructions

- You may use the language of your choice between **Go, Ruby, Python, Java, and JavaScript** (Node & TypeScript included). Please ask us if you'd like to do it in another language as we may not have reviewers corresponding.
- You can access Internet for information.
- You should favor **commonly-used libraries** whenever possible.
- Prepare **a document** (ex: README.md) describing your solution, the approach you used to solve the problem, issues faced, potential improvements, instructions about how to run your solution, etc.
  💡 *Pretend you are submitting this solution as a pull request to be reviewed by your teammates: the review will be done as a 90 min debrief interview with Qonto engineers.*
- When you are finished, please gather your files in a **.zip or .tgz archive** and submit them by email.

Time

We recommend spending **3 - 4 hours** solving the test, but if you need more time, don't hesitate to take it and explain why.

We are interested in your **thought process**, and your **problem-solving approach**, and we will consider all submissions even if incomplete/unfinished.

That said, if you'd like to take more time to improve, or "finish" your solution, feel free to do so (just let us know beforehand), and we will consider the final submission as well.

Resources at your disposal

Please download the following archive:

https://qonto-skilltest-dl.s3.eu-west-3.amazonaws.com/backend/interview-test-backend-assets-20250401.zip

You can find the following files to help you on your way:

- `sample1.json`, `sample2.json`: the bulk transfer requests you receive,

- `qonto_accounts.sqlite`: a sample SQLite database you can use.

The bulk transfer requests are JSON documents, with the following attributes:

- `organization_bic` and `organization_iban` uniquely identify the Qonto customer's account.
- Each entry in `credit_transfers` in the JSON file is a transfer.
- `amount` is the amount in Euros of the individual transfer, as a string, with optional decimal places (no more than 2). Always positive.
- `counterparty_bic` and `counterparty_iban` represent the account of the counterparty. These are accounts external to Qonto: they only exist in other banks, and do not appear in our database.
- The name of the counterparty is in `counterparty_name`.
- The description of the transfer is in `description`.

👉 The database of the bank

You can use the provided `qonto_accounts.sqlite`. It contains the Qonto customer's account and transfers linked to them.

> You can also use your own database, but in this case, you must provide a **Dockerfile** allowing us to properly run your solution, along with the means to create the necessary databases, tables, seed data, etc.

The schema of the provided `qonto_accounts.sqlite` is composed of two tables described below.

`bank_accounts`

Structure:

- `id` *INTEGER*: Unique internal identifier of customer bank accounts.
- `organization_name` *TEXT* Name of the organization that opened the account.
- `balance_cents` *INTEGER* Current balance of the account, in cents of Euros.
- `iban` *TEXT* IBAN of the account.

- `bic` *TEXT* BIC of the account.

transfers

- `id` *INTEGER* Transfer unique identifier.
- `counterparty_name` *TEXT* Name of the counterparty of the transfer.
- `counterparty_iban` *TEXT* IBAN of the counterparty.
- `counterparty_bic` *TEXT* BIC of the counterparty.
- `amount_cents` *INTEGER* Amount, in cents of Euros, of the transfer.
- `bank_account_id` *INTEGER* `id` of the related `bank_account`.
- `description` *TEXT* Description of the transfer.

Feedback

We would appreciate if you could answer the following:

1. **How much time did you spend completing this test?**
   - ☐ Less than 3h
   - ☐ 3h to 4h
   - ☐ 4h to 5h
   - ☐ 5h to 6h
   - ☐ More than 6h (your time: _____)
2. **How proud are you of your work?**
   - ☐ Very proud
   - ☐ Fairly proud
   - ☐ Good enough
   - ☐ Somewhat disappointed